METHOD AND APPARATUS FOR COMPRESSING VIDEO SEQUENCES

Related applications

The application is related to the following applications assigned to the same applicant as the present invention and filed on even date herewith, the disclosure of which is hereby incorporated by reference:

Method and apparatus for compression of video images and image residuals (Our file: IDT 019 WO)
Method and apparatus for motion estimation (Our file: IDT 020 WO)

Field of invention

This invention relates to the compression of sequences of digital video images, so as to allow more efficient transmission or storage of the sequences.

Background of invention

The PCT patent application "Method and Apparatus for data analysis", international publication number WO 95/08240, which is hereby included by reference, discloses a technique, called IDLE, for modelling video sequences, based on subspace models of motion and intensity.

The PCT patent application "Method and Apparatus for co-ordination of motion determination over multiple frames", PCT application number PCT/EP 96/01272, international publication number WO 96/29679, which is hereby included by reference, discloses a technique for co-ordinating motion estimation with subspace modelling.

These two patent applications describe techniques that can be used efficiently for modelling data, in particular video sequences. However, they do not describe a complete, simple, efficient video encoder based on the IDLE principle.

The patent application "Apparatus and Method for Decoding Video Images", PCT application number PCT/EP 95/02105, PCT international publication number WO 95/34172, which is hereby included by reference, discloses an implementation architecture for decoding an IDLE model.

Hybrid encoders for video sequence compression, based on Discrete Cosine Transform (DCT) and motion compensation exist. One example is the standard ISO/IEC 13818-2 (MPEG-2 video), which is hereby included by reference. This type of video codecs provides a good compromise between coding efficiency and implementation complexity. However, in certain application scenarios higher compression ratios achieving a similar subjective reconstruction quality are desired.

Objects of invention

It is an object of the present invention to provide a technique to reduce the amount of data required for the representation of digital video in order to achieve a high level of data compression.

It is a further object of this invention to provide an encoding technique with a simple structure amenable for cost-effective technical implementation for an encoder.

It is a further object of this invention to provide a decoding technique with a simple structure amenable for cost-effective technical implementation for a corresponding real-time decoder.

Summary of the invention

The invention is based on modelling sets of consecutive images according to the IDLE principle. Each set of consecutive images is modelled by choosing one image as reference image and representing the other images in terms of the reference image and a subspace model of motion. The sets of consecutive images may overlap. The reference image and the subspace model of motion can themselves be compressed by means of e.g. standard image compression or data compression schemes.

Various enhancements to this basic principle are possible: Residuals can be transmitted. The length of the set of consecutive images can be adaptively adjusted depending on the sequence content. The complexity of the subspace model of motion can be adjusted adaptively. Bi-directional (forward and backward) prediction can be used to improve coding efficiency or image quality. Changes in intensity can be modelled. Arbitrarily shaped video objects can be supported. Redundancy between neighbouring sets of consecutive images can be exploited. Various sampling grids, image resolutions, and aspect ratios can be supported. In particular, interlaced and progressive scanned video input as well as different colour formats can be supported.

Notations and Definitions

Instantiation: Result of a mapping.

Warping: Mapping a source to an instantiation using motion information (displacement).

Forward warping: Mapping every pixel of the source to a displaced point in the instantiation and keeping its attribute. The attribute of a pixel in the instantiation is calculated by interpolating the attributes of mapped pixels in its neighbourhood.

Backward warping: Mapping onto every pixel of the instantiation from a displaced point in the source. The attribute of a pixel in the instantiation is calculated by interpolating the attributes of pixels in the neighbourhood of its displaced point in the source.

Prediction target: To be predicted from a source.

Prediction: Mapped source as an approximation of a prediction target.

Forward prediction: A prediction, where the prediction target lies in the future of the source.

Backward prediction: A prediction, where the prediction target lies in the past of the source.

Prediction error: Difference between the original prediction target and the prediction.

Residual: Prediction error which is possibly post processed.

Reconstruction: Sum of prediction and residual.

Blending: Pixel by pixel combination of predictions.

D (Displacement in Address): Motion field.
D_v and D_h (Displacement Vertically and Horizontally): Vertical and horizontal component of motion field.

Reference image: Source to be mapped for the prediction of a target.
I-image: Intra, non predicted reference image.
P-image: Predicted reference image.

B-image: Bi-directional predicted image.
U-image: Unidirectional predicted image.

The following logical operators will be used:
$\wedge$: logical AND
$\vee$: logical OR
$\neg$: logical NOT

The term image comprises frames as well as fields, as defined in the standard ISO/IEC 13818-2 (MPEG-2 video), already included by reference. Moreover an image may contain colour information, too.

By "fidelity" is meant how well a prediction or reconstruction predicts or reconstructs an original, expressed in the root mean square (RMS) of the prediction error, or other error measures. By "quality" is meant the subjective, visual quality of the prediction or reconstruction, which in addition to fidelity takes into account how annoying various types of image artefacts are for the human visual system.

An encoder/decoder system works according to the IDLE-principle, if it
1. determines subsets of consecutive images (SCI) whose union covers the whole sequence,
2. extracts relevant information from an SCI to build an IDLE-model,
3. combines the information contained in the IDLE-models to build a prediction of the original sequence.

An IDLE-model for an SCI of length N consists of
1. one reference image
2. a set of m subspace models for m different attributes ($m \geq 0$),

where the i -th subspace model consists of
1. a set of k_i basis images containing data of attribute i $(1 \leq k_i \leq N-1)$
2. a set of $N-1$ temporal weight coefficient vectors of length k_i .

The combination of the IDLE models is performed using blend-fields.
The attributes may be components of motion-(displacement-) vectors or intensity etc.

The encoder/decoder system described in the following can work using two different motion compensation schemes, a forward and a backward one. Within the forward motion compensation scheme the sources (e.g. reference images) are warped to the targets by forward

warping wherein the motion fields are defined in the source positions. In the backward motion compensation scheme the sources (e.g. reference images) are warped to the targets by backward warping wherein the motion fields are defined in the target positions. In order to describe both schemes two abstract modules/operators are used: Warp and OppositeWarp. In the forward motion compensation scheme Warp performs a forward warping and OppositeWarp a backward warping whereas in the backward motion compensation scheme Warp performs a backward warping and OppositeWarp a forward warping. In the description a distinction between forward and backward compensation scheme will not be made unless necessary.

Brief description of the figures

Fig. 1a,b show the encoder in its basic form;
figs. 2 a - d illustrate the concept of subspace modelling;
fig. 3 illustrates a decoder in its basic form;
fig. 4 a, b show the codec containing a compression module;
fig. 5 illustrates the context for updating a subspace model of motion;
fig. 6 illustrates using the first image in each set of consecutive images as reference image;
fig. 7 illustrates using the centre image of each set of consecutive images as reference image;
figs. 8 a - d illustrate adaptive determination of the length of a set of consecutive images;
fig. 9 illustrates the context for determining rank of the subspace model of motion;
figs. 10 a - d illustrate the motivation for using an extended reference image;
figs. 11 a - h illustrate one method for estimating an extended reference image;
fig. 12 illustrates the use of P-images;
figs. 13 a - c illustrate the need for bi-directional prediction;
fig. 14 illustrates the use of B-images;
fig. 15 a - c shows the encoding using bi-directional predictions;
fig. 16 shows a decoder adapted to bi-directional prediction;
figs. 17 and 18 show two possible contexts for calculation of residuals;
fig. 19 shows how residual dampening may be integrated into the system;
fig. 20 illustrates the context for subspace modelling of intensity;
figs. 21 a - b show how subspace modelling of intensity changes can be implemented into an encoder;
figs 22 a – b show how to predict images under presence of a subspace model of intensity changes;
fig. 23 illustrates preconditioning of intensity changes before subspace modelling;
fig. 24 a - c show how to exploit redundancy of subspace models for consecutive SCI's.

First preferred embodiment

Fig. 1a shows the invention in a first preferred embodiment. An encoder Enc 100 receives as input a sequence of video images 101 and transmits a representation of the sequence as an IDLE model 104. The encoder has an image store ImageStore 110 buffering a certain number of images of the incoming sequence.

The encoder has a module BuildMod 120 for building an IDLE model. It is built from images read from the image store ImageStore 110.

The module TraMod 130 transmits the IDLE model.

The module BuildMod 120 will now be described with reference to fig. 1b. BuildMod 150 reads as input the set of consecutive images 151, consisting of a number of images N. In this preferred embodiment the number of images N is fixed, e. g. 15. One of the images in the set of consecutive images 151 is selected as a reference image. This can be the first image. A switch 160 sends this image into a reference image store RefImgSto 170. The switch 160 sends each of the other images to a motion estimator ME 180. Motion is estimated between the reference image and the other images. This results in $N-1$ motion fields.

The motion estimator ME 180 can be based on several principles. One possibility is to use a block based motion estimator, finding, for each block of pixels in an image, a motion vector to a similar block of pixels in another image.

Another possibility is to use a gradient-based method for calculating optical flow, as described in "Determining Optical Flow", B. K. P. Horn and B. Schunck, "Artificial Intelligence" 17, 1981, which is hereby included by reference.

Yet another possibility is to use mesh-based motion fields, as described in "An H.263-compatible system for video coding based on a triangular mesh", Y. Altunbasak, A. Murat Tekalp, International Picture Coding Symposium, 1996, which is hereby included by reference.

Yet another possibility is to use the system as described in the patent application "method and apparatus for motion estimation" already included by reference.

The motion fields are sent to a module BuildSSMod 190 that builds a subspace model of the motion.

The collection of the reference image RefImg 151 and the subspace model of motion MotionSSMod 152 together constitute an IDLE model 153.

The concept of a subspace model, as produced by BuildSSMod 190, will now be described with reference to figs. 2 a to d.

A given data matrix A of size $m \times n$ can be factorised into the product $A = T \cdot P$, where the matrices T and P have dimensions $m \times m$ and $m \times n$, respectively. The individual elements $a_{i,j}$ of A can explicitly be written as

$$a_{i,j} = \sum_{k=1}^{m} t_{i,k} \cdot p_{k,j} .$$

The m rows of the matrix P constitute a set of basis vectors for the m-dimensional vector space spanned by the rows of A. The matrix T indicates the linear combination of the basis vectors in P to rebuild the data matrix A.
The collection of the k-th column of the matrix T and the k-th row of the matrix P is called the k-th factor.

In case that the number of factors in the sum is limited by $f < m$ a residual matrix E of dimension $m \times n$ must be added to fully reconstruct the matrix A. Hence the individual elements $a_{i,j}$ of A read

$$a_{i,j} = \sum_{k=1}^{f} t_{i,k} \cdot p_{k,j} + e_{i,j}$$

The number f of selected factors will be referred to as the rank of the model. The collection of the $f < m$ rows of the matrix P constitute a set of basis vectors for an f-dimensional vector space. This vector space is a subspace of the space spanned by the rows of A. Hence we will call it a subspace model.
The case $f < m$ is now illustrated with reference to figure 2a. The given data matrix A 200 of size $m \times n$ is represented by $A = T \cdot P + E$, where the matrices T 202, P 204 and E 206 have dimensions $m \times f$, $f \times n$ and $m \times n$ respectively.
The n-th row a_n 210 of the matrix A 200 is rebuild according to fig. 2b where the pertaining n-th row t_n 212 of the matrix T 202 is multiplied with matrix P 214 and the pertaining n-th row e_n 216 of the matrix E 206 is added.

There exist several methods for computing such a factorisation of the matrix A 200. Some methods are described in H. Martens and T. Naes, "Multivariate Calibaration", chapter 3 (John Wiley & Sons, 1989), which is hereby included by reference. In particular, Principal Component Analysis and Karhunen-Loeve transform can be used as well as the Singular Value Decomposition, in short SVD. Algorithms and theory for computing the SVD can be found in G. H. Golub and C. F van Loan "Matrix Computations" (John Hopkins University Press, Baltimore, 2nd Edition, 1989), which is hereby included by reference.

Further information regarding subspace modelling of motion can be found in the patent application "Method and Apparatus for Data Analysis", international publication number WO 95/08240, already included by reference

The matrix A 200 may stand for any kind of data. In the current embodiment A is used to apply the above described subspace modelling technique to motion data. One row of the matrix P 204 represents a basis image of motion corresponding to one factor. One column of the matrix T 202 contains the temporal weight coefficients corresponding to one factor. The collection of all retained factors will be referred to as a subspace model of motion.

Motion data may consist of one or more motion fields, which e.g. represent computed motion for a set of consecutive images.
Motion fields generally consist of one motion vector for each pixel or for each group of pixels. In 2-dimensional images, the motion fields have one vertical and one horizontal component, which are denoted by D_v and D_h, respectively.

The vertical and the horizontal components of a motion vector can be considered as a real and imaginary part of a complex valued vector field. Hence, the modelling can be performed using complex SVD algorithms or similar methods. The motion fields can also be represented in terms of magnitude and angle. One subspace model for magnitude and one for angle can be used.

The arrangement of the motion data in matrix form can be done in several ways:
Each single component can be treated independently as shown in fig. 2c; this is called separate modelling.
E.g. the vertical component of a motion field, D_v 220, having size n_v vertically and n_h horizontally, can be strung out to be represented by a row vector 222 of length $n = n_v \cdot n_h$. Each of the constructed vectors represents one row of the data matrix A 200.

The same procedure can be applied on the horizontal component of a motion field D_h.
The result is a separate subspace model for each, the vertical and the horizontal component of the motion data.

Alternatively, as illustrated in fig. 2d, the vertical and horizontal component of the motion fields D_v 230 and D_h 232 can be strung out as one row vector 234 jointly, which then has the length $n = 2 \cdot n_v \cdot n_h$. This leads to one subspace model of motion describing motion in both the vertical and horizontal dimension. This is called joint modelling.

In the following, such a joint or separate subspace model covering both components of motion fields (e.g. vertical and horizontal) will be referred to as a subspace model of motion. A distinction between joint and separate modelling will not be made unless necessary.

The rank f can be chosen to be significantly lower than the number of images in the corresponding set of consecutive images. Also, for many video sequences, the motion residual E 206 will contain small values, or can be dropped without too much degradation in reconstruction quality. In total, this gives a potential for compression, in that a low number of temporal weight coefficients and basis image vectors, T 202 and P 204, and optionally the motion residual matrix E 206, can be transmitted instead of the full motion fields A 200.

The rank can be set to a constant. E.g., for a set of consecutive images size N = 15 images, the rank can be set to $f = 3$.

One simple implementation of TraMod 130 is to just transmit the data on a given communication channel.

An IDLE decoder corresponding to the encoder described above will now be described, with reference to fig. 3. The decoder Dec 300 receives an IDLE model 301 with its module ReceiveMod 310 and sends it to a module Predict 320 which provides a prediction 302 of each of the original images within the set of consecutive images.
Within the module Predict 320 the reference image part of the IDLE model is sent to the reference image store RefImgSto 330. The subspace model of motion is sent to the module RebuildMotion 340, where motion fields are rebuilt from the subspace model. The rebuilding of the motion field corresponding to a certain image is performed by calculating a linear combination of the basis images using temporal weight coefficients pertaining to that image. The module Warp 350 uses the reference image and the rebuilt motion fields to provide the

predictions 302 of the images within the set of consecutive images. The reference image itself passes the module Warp 350 unchanged.

The above procedure is applied for each IDLE model.

The Warp module 350 warps the image content of a reference image according to a motion field. Several methods for doing this are given in "Digital Image Warping", third edition, George Wolberg, ISBN 0-8186-8944-7, IEEE Computer Society Press, 1994, which is hereby included by reference.

A related method is to interpolate motion between motion vectors computed by block matching, as described in "Feedback Loop for Coder Control in a Block-Based Hybrid Coder with Mesh-Based Motion Compensation", Jörn Ostermann, Proceedings of ICASSP '97, Munich, IEEE Computer Society Press 1997, which is hereby included by reference.

Another method is given in the PCT patent application "Apparatus and Method for Decoding Video Images", international publication number WO 95/34172, pages 31-43, which is hereby included by reference. A full description of an IDLE decoder is given in the same patent application.

Second preferred embodiment

The IDLE model produced by the encoder may be compressed.

The reference image, the temporal weight coefficients and basis images of the motion model can be compressed before being transmitted in module TraMod 130 from fig. 1.
Such an encoder scheme is shown in 4a. The encoder Enc 400 receives as input a sequence of video images 401 which are buffered in the module ImageStore 410. The module BuildMod 420 reads SCIs from ImageStore 410, builds the corresponding IDLE models and delivers the models to the module CompMod 425. The module CompMod 425 compresses the models. The compressed models are then transmitted by the module TraMod 430.
The pertaining decoder will now be described with reference to fig. 4b. The decoder Dec 450 receives an IDLE model 451 with its module ReceiveMod 460 and sends it to a module DecompMod 470 which decompresses the model. The decompressed model is sent to the module Predict 480 which provides a prediction 452 of each of the original images within the set of consecutive images. Module Predict 480 works as module Predict 320 in fig. 3. The above procedure is applied for each IDLE model.

In addition to the IDLE model the residual matrix E 206 may be also compressed and transmitted to the decoder. For all those types of data various compression methods can be used.

The reference image is a still image, and may be compressed using one of many still image compression methods available. In particular, transform coding may be used. One such method, based on the Discrete Cosine Transform, DCT, is JPEG, as described in "JPEG Still Image Compression Standard", William P. Pennebaker, Joan L Mitchell, ISBN 0-442-01272-1, 1993, which is hereby included by reference. Other methods are vector quantisation, predictive coding, fractal coding, wavelet based transform coding and other still image coding techniques as mentioned in "Digitale Bildcodierung", Jens-Rainer Ohm, Springer Verlag 1995, which is hereby included by reference.

Each basis image usually has some characteristics in common with conventional still images; in particular, there is usually spatial redundancy. Hence each basis image can be compressed using any still image compression tool mentioned above with an adaptation to match the dynamic range pertaining to the different types of data.

The basis images may be represented in low-resolution.
The corresponding sub-sampling or up-sampling may be done as part of the compression and decompression.

Alternatively, the basis images may be represented by adaptive meshes (wire-frame) which can be generated by e.g., the Delauney triangulation process.

Other types of irregular sampling grids may also be used for the efficient representation of the basis images.

Various matrix factorisation processes can produce scaled basis images suitable for the used set of compression tools. One method e. g. is that the basis images can be adjusted so that the maximum impact of the quantisation error will be the same for all factors, if the characteristic of the quantiser is the same for all factors. Alternatively, appropriate scaling factors for the temporal weight coefficients as well as for the basis image for a given factor can be calculated and transmitted individually.

Another method is to add or subtract an offset value to the elements of a basis image, and scale the basis images to fit into an appropriate dynamic range. The scaling factors as well as the offsets need to be transmitted additionally to the decoder. Instead transmitting the scaling factors

directly, they can be used to adjust the quantiser step size. This method may also be applied to all basis images simultaneously.

The temporal weight coefficients for the first factors often have high level of temporal correlation. Hence predictive coding techniques may be applied for their compression. This includes compression methods which use all types of frequency transforms. Examples for frequency transforms are the family of discrete cosine/sine/Fourier transforms as well as various wavelet transforms.

Third preferred embodiment

In this embodiment, the subspace model of motion is updated recursively.

In the first preferred embodiment, the motion fields are estimated for each image of the set of consecutive images. Subsequently, the subspace modelling is performed.

In some applications, memory consumption may be important. It is advantageous to have a method that does not need intermediate storage for all motion fields. It may be important to have parallel algorithms, so that motion estimation and calculation of subspace models can be performed concurrently. For such applications, the subspace models can be updated each time a new image is processed instead of being built at the end of a set of consecutive images.

This is illustrated in fig. 5. The BuildMod module 500 again takes a set of consecutive images 501 as input, a switch 510 sends one image to a reference image store RefImgSto 520 and the other images to a motion estimator module ME 530. The resulting motion fields from ME 530 are now sent to a module for updating a subspace model, UpdateSSMod 540. At the start of a set of consecutive images, a model store StoreSSMod 550 for a subspace model is initialised to zero. At each new motion field, UpdateSSMod 540 reads the current subspace model from StoreSSMod 550, updates the subspace model, and stores it back.

Several methods for updating a subspace model exist. Some are given in "Multivariate Calibration", H. Martens & T. Naess, John Wiley & Sons, 1989, already included by reference. Other methods can be found in:
R.D.DeGroat, "Noniterative subspace tracking". IEEE Trans.Signal Processing, Vol.40, pp. 571-577, March, 1992, and in:
E.M.Dowling, L.P.Ammann, R.D.DeGroat."A TQR iteration based adaptive SVD for real-time angle and frequency tracking". IEEE Trans.Signal Processing, Vol42, pp. 914-926, April, 1994, which are hereby included by reference.

Fourth preferred embodiment

In this embodiment, the position of the reference image within an SCI can be determined to satisfy various performance criteria.

In the first preferred embodiment, the first image of each SCI was selected as the reference image. Fig. 6 illustrates the resulting structure for the ordering of the images pertaining to an IDLE model 600. The first image in the set of consecutive images, denoted by I 610, is used as a reference image. The other images, denoted by U 620, 630, 640, 650, are represented as forward predictions of the reference image. The corresponding motion fields are modelled using subspace modelling as described above.

Other images may be chosen as reference images. Fig. 7 shows a centre image 730 of a set of consecutive images pertaining to an IDLE model 700. Earlier images 710, 720 are backward predicted versions and later images 740, 750 are forward predicted versions of the centre image. The corresponding motion fields can be modelled as one motion model. Alternatively, one subspace model can be used for the earlier images 710, 720 and one separate model can be used for the later images 740, 750.

The most representative image in the set of consecutive images may serve as the reference image in order to enable an optimum prediction using the pertaining IDLE model.

The selection of the reference image can be based on motion compensated predictions, e.g., motion estimation can be performed between the first image in the set of consecutive images and each of the following images as far as the corresponding prediction satisfies a fidelity criterion, a quality criterion or a cost (in terms of bits) criterion. The original image corresponding to the last satisfying prediction is chosen as the reference image.

Alternatively, the reference image can be determined by computing an average or median histogram over intensity or colour for all images, and then selecting the image whose histogram has the highest similarity with an average or median histogram.

Selecting a suitable image to be used as reference image for a set of consecutive images can also be done using analysis by synthesis. Several images can be tried as the reference image, and the selection that results in the most efficient modelling can be accepted.

Fifth preferred embodiment

In this embodiment of the invention, the number of images to be included in one set of consecutive images is determined adaptively by BuildMod 420, referring back to fig. 4a.

For related or similar motion fields, subspace modelling can bring advantages, e.g. with regard to compression ratios. However, if motion fields with little similarity are modelled together, the subspace modelling may bring disadvantages regarding fidelity vs. compressibility, both for motion subspace models and the overall system.

BuildMod 420 determines the number of images to be included in one set of consecutive images. To this end, the next image to be processed is read from the ImageStore 410. Motion estimation calculates the motion field between the reference image and the current image. A fidelity criterion is used to decide whether this motion field will be included in the process of building the current subspace model.

If an image is not taken into account for the current subspace model, BuildMod 420 passes the current IDLE model to CompMod 425. In the case where the subspace model of motion is updated recursively, as illustrated in fig. 5, the subspace model in StoreSSMod 550 and UpdateSSMod 540 additionally must be re-initialised to empty before building the next model.

Several methods can be used for the determination of the length of a set of consecutive images.

Motion fields often have little similarity across scene shifts. One way to determine the length of the set of consecutive images is therefore to use a scene shift detector. Several methods for detecting scene shifts exist. One is to detect differences between histograms of pixel values for two or more consecutive images. Another is to detect scene shifts by measuring prediction errors.

Other methods for defining sets of consecutive images use motion estimation as described now with reference to fig. 8a-d.

Fig. 8 a shows how the motion estimation module ME 530 from fig. 5 or 180 from fig. 1b can be used for defining sets of consecutive images, in addition to its normal function. The functions of switch 810, RefImgSto 812, ME 814, are the same as described in fig. 5. The motion field, resulting from ME 814, is used to warp the reference image in the module Warp 816 to produce a predicted image. The warping is performed in the same direction (forward or backward) as in the decoder.

The absolute value of the difference between the predicted image and the original image is summed up in SumAbsDiff 818. For related images, this will often give a relatively small result, while for unrelated images, this will often give a large result. The sum is fed to the encoder control EncCont 819, where it can be compared with a given threshold. As long as the sum stays below the threshold, the motion field is passed to BuildSSMod 820 so that it is included in a subspace model of motion. If the sum is larger than the threshold, the end of the set of consecutive images is reached. The reference image 821 and the subspace model of motion 822, together constituting an IDLE model 823, are given as output.

Figure 8b illustrates an alternative method, which uses the module OppositeWarp 836, where the warping is performed in the opposite direction as in the decoder. The functions of switch 832, RefImgSto 834, ME 835, BuildSSMod 837 are the same as described in fig. 8a. The current image is warped in module OppositeWarp 836 using the motion field estimated by ME 835. The resulting prediction is expected to be similar to the reference image for the case of highly related images. Therefore, the result from a module 838 for summing absolute differences between the warped image and the reference image can be compared against a given threshold in the encoder control EncCont 839. This way the start of a new set of consecutive images may be triggered.

The similarity between an original image and the prediction of the image can also be expressed in terms of spatial size of prediction failure areas. One possibility is to calculate an intensity residual image as the difference between the original image and the prediction, divide the residual image into spatial blocks, and then count the number of spatial blocks where the average error is above a given threshold. A spatial block denotes a group of pixels or one pixel. The average error can be found by the sum of absolute values of the spatial block divided by the number of pixels in the spatial block, or by root mean square (RMS), or by another fidelity criterion. Other characterisations can also be used, like a median value. A threshold can be used to distinguish between spatial blocks, which exhibit a prediction failure or prediction success. Another threshold can be used for the allowed number of spatial blocks designated as prediction failure. The end of the set of consecutive images is not reached as long as the number of spatial blocks with prediction failure stays below a certain threshold, e.g. 5 % of the total number of spatial blocks in the image.

When defining spatial blocks with prediction failures, characteristics of the human visual system may be considered to get an estimate of how visible artefacts will be. One possible method is to slightly blur the residual image before thresholding. Another possible method is to process the thresholded image with morphological filters like 'Opening' and 'Closing'. Also methods from "Method and apparatus for compression of video images and image residuals", already included

by reference, may be used to discriminate between highly and less visible artefacts. Residual values which correspond to invisible artefacts may be removed from the residual image before each spatial block is characterised as a prediction success or a prediction failure.

This is illustrated in fig. 8 c. A module BuildMod 850 for building a model is shown. The functions of switch 860, EncCont 869, RefImgSto 861, ME 862, OppositeWarp 863 and BuildSSMod 864 are the same as described for fig. 8 b.

In addition, there is a module DefBounds 866 for defining bounds wherein the artefacts are declared to be invisible. This module can exploit masking effects of the human visual systems. Examples include entropy masking and non-linear response for intensity. It produces an upper and a lower bound for each spatial block. A more detailed description is given in "Method and Apparatus for compression of video images and image residuals", already included by reference.

There is also a module RelToBounds 867. It relates for each pixel or block of pixels the difference between the original image and the predicted image, given as output of the difference module 865, to the bounds. It can be based on several principles. One is to check whether the residual is within the bounds, producing a yes/no result, e.g. encoded as 0 if the residual is within the bounds and 1 otherwise. Another principle is to compute the ratio between the residual and the upper or lower bound. Optionally, if the ratio is smaller than 1, then it can be set to zero. Yet another method is to compute the difference between the residual and the upper or lower bound.

A summation module 868 then sums up the results of RelToBounds 867 and gives the sum to EncCont 869, which can make a decision about the length of the set of consecutive images based on this sum.

The modules difference module 865, DefBounds 866, RelToBounds 867 and summation module 868 may also be used within the structure according to fig. 8a instead of SumAbsDiff 818, where DefBounds and the difference module get as input the current image instead of the reference image.

Alternatively, the fit of a new motion field to a subspace model can be used for deciding whether to start a new set of consecutive images. This is illustrated on fig. 8 d, showing a module BuildMod 875. The result from ME 884 is projected on a subspace model of motion in ProjOnSSMOD 886, producing a residual Res, which can not be described within the subspace model. If the sum of absolute values of the residual, as computed in SumABS 888, is above a threshold, then EncCont 874 may decide to start a new set of consecutive images. Otherwise,

UpdateSSMod 892 updates the subspace model of motion stored in StoreSSMod 890. When the result of SumAbs 888 is small, the motion field fits well into the subspace model.

The threshold can be absolute, e.g. corresponding to an allowed average or maximum deviation, expressed in the number of pixels, between the motion field as estimated by ME 884 and the same motion field as modelled by BuildSSMod.

Some methods for updating a subspace model have a projection of new data on old factors as one of the first steps. ProjOnSSMOD 886 can therefore be seen as a first step of UpdateSSMod 892.

In some applications, random access into the video sequence is required. Examples include channel change in connection with broadcasting, or fast forward mode for storage based systems. In such systems, some maximum temporal distance between reference images is prescribed. An encoder control module EncCont can then count the processed images, and force reference images at given intervals by starting a new set of consecutive images. This can be combined with any other method for defining the length of a set of consecutive images. E.g., a system could have an image counter. Each time a new set of consecutive images is triggered using any of the methods outlined in figs. 8 a-d; the image counter is reset to zero. For each processed image, the image counter is increased by one. If the image counter reaches a given value, a new set of consecutive images is triggered and the image counter is reset to zero. E.g., for a TV system using 25 images per second and where other components than IDLE modelling contribute with a delay corresponding to at most 5 images, the threshold may be set to 20 if random access with at most one second delay is prescribed.

Sixth preferred embodiment

The concepts of adaptive determination of the length of a set of consecutive images and the adaptive choice of a reference image can be combined.

One method is to first determine the length of the set of consecutive images according to any of the described methods using a preliminary reference image, and then choose the final reference image adaptively as described in the third embodiment.

Another method is based on selecting the length for a set of consecutive images and choosing the reference image simultaneously. The goal is a representation as illustrated in fig. 7, such that the set of consecutive images will consist of as many images as possible. For that first search for an image I 730, to be used as a reference image, that is as far away from the first image 710 as

possible while still maintaining the possibility to well predict one from the other. The search is done using motion estimation. All images between image 710 to the found reference image I 730 are included into the set of consecutive images. Then find the end of the set of consecutive images, such that all the images after I 730 can still be well predicted from I 730 as described in the fifth embodiment.

Seventh preferred embodiment

Different sets of consecutive images may have a different optimum number of motion factors f, also called the rank of the subspace model. E.g., when the images of a set of consecutive images only show a still image without motion, then zero motion factors are needed. On the other hand, when a long set of consecutive images contains complex, but highly systematic motion, like a human head talking, a subspace model that captures much of the motion needs several factors, that is, it requires a model with higher rank.

In principle, motion information is lost in several stages in the described system resulting in a loss of fidelity of the subspace model and therefore a loss of prediction fidelity. First, motion estimators will in general not be able to estimate the "true motion". Second, for a reduced rank of the subspace model, some of the motion vectors will not be fully represented by the subspace model. Third, when the subspace model is compressed, more motion information is lost unless a loss-less compression technique is used. This preferred embodiment illustrates aspects of making a compromise between compression ratios and fidelity or quality of the motion estimation after rank reduction.

One way of reducing the rank, is to first update or build a subspace model using a higher number of factors, and then determining a final rank after all images of the set of consecutive images have been processed. This is illustrated in fig. 9. BuildMod 900, corresponding to the module 420 with the same name in fig. 4a, has a module UpdateSSMod 950 for updating a subspace model stored in StoreSSMod 960. Before the subspace model is passed on from BuildMod 900, the rank of the model is determined in DetRank 970 and factors may be deleted.

Several methods for rank determination exist.

One simple assumption is that a long set of consecutive images may have more independent motion phenomena than a short set of consecutive images. DetRank may then be based on a set of rules or a table giving the number of factors f to be used depending on the length N of a set of consecutive images. E.g. such a set of rules could be:

$$f = \begin{cases} 0 & \text{for } N < 2 \\ 1 & \text{for } 2 \le N < 5 \\ 2 & \text{for } 5 \le N < 11 \\ 3 & \text{for } 11 \le N \end{cases}$$

Alternatively, DetRank 1070 may be based on the content of the subspace model.

If SVD is used for subspace modelling, one method is to discard those factors that have singular values below a threshold. Another method is to define a relative threshold. E.g., if the singular value of a factor has less than 5 % of the largest singular value of a subspace model, then the factor is discarded.

Another method is to define a smallest relevant motion. E.g., one may define that the impact of motion smaller than one half pixel is not visible. Therefore, cut away those factors whose largest absolute value of the temporal weight coefficients multiplied with the largest absolute value of the basis image does not exceed the value of 0.5.

The image content may have an influence on the necessary fidelity of the motion subspace model and therefore on its rank. E.g., in areas without texture of the image some "false" motion may be of low visual importance, while in textured parts of the image even small errors in motion may be visible.

To determine the importance of certain factors one may characterise allowable deviations of the motion for different parts of the image based on the image content. If the prediction based on the motion represented by the subspace model without a certain factor stays within these allowable deviations, then this factor is deleted. The allowed deviation may be derived using various confidence measures. A number of methods to determine confidence measures are described in the patent application "Method and apparatus for compression of video images and image residuals", already included by reference.

Reducing the rank can alternatively be regarded as a part of the compression of the subspace model of motion. During compression of a subspace model of motion, each basis image can be coded separately using an image-compression tool. This tool can be controlled so that only significant parts of the basis image are transmitted. E.g., if the image-compression tool is based on dividing the basis image into spatial blocks and transform coding each block using DCT, then it can be checked if the motion related to this factor and block is below a given threshold, say, one half pixel. If it is the case, then the block may be skipped, otherwise it may be coded. The

number of factors may be reduced according to the number of non-skipped spatial blocks in the pertaining basis images.

Alternatively, the rank of the model may be reduced by considering the singular values, or the sum of squares of basis images or temporal weight coefficients. Often, using SVD for subspace modelling, the first factors have mainly to do with "signal", and the last factors have mainly to do with "noise". The "noise" may include e.g., errors originating from the motion estimation process or from thermal noise in the image. Often, the factors mainly related to "signal" have strongly decreasing singular values, while the factors related to "noise" will have a much flatter distribution of the singular values. The transition from strongly decreasing to flatter distribution can often be seen as a "knee point" on a plot of the singular values as a function of the factor numbers. An encoder can determine an appropriate knee point and discard all factors with singular values smaller than the singular value at the knee point.

The mechanisms described for DetRank 1070 may be performed as part of UpdateSSMod 1050 for every update operation. This may have advantages in terms of computational complexity or storage requirements due to the fact that UpdateSSMod 1050 and StoreSSMod 1060 only need to consider the selected number of factors.

The methods for determining an intermediate value for the rank in UpdateSSMod 1050 and the final determination in DetRank 1070 may be combined. The rank may be reduced from full to intermediate in UpdateSSMod 1050, mostly in order to save resources, while the rank may be further reduced in DetRank 1070, mostly to optimise compression ratios.

Alternatively, the above described methods for rank reductions may be spread out to different locations in the encoding process to achieve a multi-stage processing scheme. The reason for this is to add flexibility to the rank determination in order to take into account the potentially changed importance of factors throughout the encoding process.

All of the above methods for determining the rank apply both to the case of joint modelling, with one subspace model of motion covering both vertical and horizontal dimension, and separate modelling, where there is one subspace model for vertical dimension and another for the horizontal.

Furthermore, for separate modelling the rank can be determined in common for both dimensions, or the rank can be determined independently for each dimension.

Eighth preferred embodiment

In some applications only defined areas of the images need to be coded. The shape information for those areas may be selected by a manual or an automatic segmentation process. The shape information designates the set of valid pixels in an image. In this context the notion of 'video objects' is commonly used, e.g. within the context of the MPEG-4 standardisation body.

The shape information of video objects is often given in terms of a so-called 'alpha channel'. In this patent application the term 'shape field' will be used. Binary shapes may be used as well as grey-level shapes. The MPEG-4 video coding toolbox describes numerous methods for handling shaped video objects. Examples are the shape-adaptive DCT and various padding methods in the context of block-based video compression.

In the patent application "Method and Apparatus for Data Analysis", already included by reference, a method for building subspace models for shaped video objects is described. The patent application "Method and Apparatus for Motion Estimation", already included by reference describes methods to estimate motion in the context of shaped video objects.

Shape information may be defined only for a reference image. The shape information for the remaining images in an SCI is then generated by a prediction from the shape assigned to the reference image using motion information. To this end the shape of the reference image is warped to the position of the respective image using the motion fields.

Alternatively, the shape information may be defined for each image. The shape information for the reference image in conjunction with the estimated motion information for the images in the SCI is used to determine a prediction model for the shape including a prediction error for the shape. The shape residual can be formed by an arithmetic difference between original and predicted shape. In this case the residual of a binary shape will be a tri-level image. Alternatively, the residual of a binary shape can be formed using an exclusive-or operation between original and prediction, in which case it will be a bi-level image (binary image).

Without using predictions the shape information for all images can be coded separately using e.g., shape coding tools from MPEG-4.

In order to obtain a high gain of compression performance loss-less as well as lossy compression methods for shape fields may be applied. One example method for coding shapes is: JBIG, as described in "A simple and efficient binary shape coding technique based on bitmap

representation", F. Bossen and T. Ebrahimi, Proceedings of ICASSP '97, IEEE Computer Society Press, 1997, which is hereby included by reference.

Several shaped video objects may be encoded independently. During the decoding process the individual shaped video objects may be composed into common images making use of composition information such as depth layering or spatial as well as temporal position for the individual video objects.

Ninth preferred embodiment

In general, it is not assured that all parts of all images can be predicted from one reference image using available motion data. Some images may have parts that are not covered by warping the reference image. This is illustrated in figs. 10 a - d. Fig. 10 a shows a reference image. Fig. 10 b shows a predicted image as a reference image warped with an arbitrary motion field. There are disappearing parts 1040, 1041, 1042, 1043 of the reference image. These are in general of little importance. However, there are also parts 1030, 1031, 1032, 1033 of the image that are not covered by the corresponding prediction.

One way to cope with uncovered parts is to transmit a residual for those parts. One can choose to handle the uncovered parts like other model failure regions. This has an advantage with regard to algorithmic simplicity.

Another way is to create an extended reference image, containing the reference image plus new content. This is illustrated in fig. 10 c. To the same reference image as was shown in fig. 10a more pixels 1060, 1061, 1062, 1063 have been added, thus forming an extended reference image. The warped reference image covers now the whole image to be reconstructed. An illustration is depicted in fig. 10 d. The subspace model of motion must be adjusted correspondingly. 1080-1083 are reconstructed areas corresponding to 1060-1063.

One method for creating an extended reference image is based on warping pixels from the original image corresponding to the uncovered pixels in the predicted image into the position of the reference image. This method will now be described in more detail, with reference to figs. 11 a-h. An original reference image I_r 1100 is given, depicting a first tree 1102 and a person 1104. An image I_n 1110 is also given, depicting the same person 1112 but a second tree 1114, due to camera panning. A motion field D 1120 is calculated between I_r 1100 and I_n 1110. In the case of forward motion compensation the motion field D is given in the position of the reference image I_r whereas in the case of backward motion compensation the motion field D is

given in the co-ordinate system of the image I_n. Fig. 11c shows the forward motion compensation case. One of the motion vectors 1122 for the tree and one of the motion vectors 1124 for the person in the reference image has been drawn. The task is to prepare a new reference image with space for both instances of the person 1104, 1112, the first tree 1102, and the second tree 1114.

An extended version 1130 of I_r (horizontal size n_h, vertical size n_v) is created, so that it contains I_r 1100, with additional space around it so that pixels from I_n 1110 can be filled in. One conservative method comprises the following steps:

1. Warping of the shape S_r of the reference image to the position of I_n resulting in the shape $S_r' = \text{Warp}(S_r, D)$.
2. Determination of the uncovered areas of image I_n defined by the shape S_u using S_r' and the shape S_n of the image I_n: $S_u = S_n \wedge \neg S_r'$. Note that S_r and S_n may consist completely of ones.
3. Finding the upper limit u_u, the lower limit d_u, the left limit l_u, and the right limit r_u of the shape S_u.
4. Determination of the minimum value $D_{v,\min}$ and the maximum value $D_{v,\max}$ of the vertical component of the motion field D as well as of the minimum $D_{h,\min}$ and maximum $D_{h,\max}$ value of the horizontal component. Note that the origin of the co-ordinate system lies in the upper left corner of the image as usual in computer graphics.
5. Determination of the limits of the uncovered areas in reference position due to the motion field (the index r denotes reference position):
   In the case of forward motion compensation:
   $u_{ur} = u_u - D_{v,\max}$; $d_{ur} = d_u - D_{v,\min}$; $l_{ur} = l_u - D_{h,\max}$; $r_{ur} = r_u - D_{h,\min}$
   In the case of backward motion compensation:
   $u_{ur} = u_u + D_{v,\min}$; $d_{ur} = d_u + D_{v,\max}$; $l_{ur} = l_u + D_{h,\min}$; $r_{ur} = r_u + D_{h,\max}$
6. Finding the upper limit u_r, the lower limit d_r, the left limit l_r and right limit r_r of the shape S_r and setting $u = \min(u_{ur}, u_r)$; $d = \max(d_{ur}, d_r)$; $l = \min(l_{ur}, l_r)$; $r = \max(r_{ur}, r_r)$.
7. Determination of the resulting horizontal t_h and vertical t_v translation of the upper left corner of the reference image as well as of the resulting extended horizontal $n_{h,E}$ and vertical $n_{v,E}$ size (the index E denotes extension):
   $t_h = -\min(0, l)$; $n_{h,E} = \max(n_h, r) - t_h + 1$;
   $t_v = -\min(0, u)$; $n_{v,E} = \max(n_v, d) - t_v + 1$

8. Extending (filling with zeros) I_r and the corresponding shape S_r with t_v pixels on the upper, with t_h pixels on the left, with $n_{v,E} - n_v - t_v$ pixels on the lower and with $n_{h,E} - n_h - t_h$ on the right border resulting in I_{re} 1130 and S_{re} (the index e denotes extended fields without further modifications).
9. In the case of forward motion compensation: Extending the motion field D 1120 in the same manner producing a motion field D_e 1140.

The extending may imply a change of the co-ordinate system. E.g., if the origin of I_r was in the upper left pixel and the origin of the extended image shall also be in the upper left pixel, then values corresponding to the extension are added to the motion field.
In the case of forward motion compensation t_h is subtracted from each point of the horizontal and t_v is subtracted from each point of the vertical component of the motion field. In the case of backward motion compensation the values t_h and t_v are added to the corresponding motion fields.

In the case of forward motion compensation the extended regions of the motion field are extrapolated from the given motion field. The extrapolation should be done so as to minimise discontinuities, both between extended and original areas and inside extended areas. One useful technique is the following diffusion process, performed on the motion field D_e:

1. Set $S_{old} = S_{re}$ and $D_{old} = D_e$.
2. For each valid pixel p (i.e. $S_{old}(p) = 1$) in the motion field, set
   $S_{new}(p) = S_{old}(p)$
   $D_{new}(p) = D_{old}(p)$.
3. For each unvalid pixel p (i.e. $S_{old}(p) = 0$) in the motion field, calculate a new shape S_{new} and a new motion field D_{new} according to:

$$N(p) = \sum_{q \in Neigh(p)} w(q,p) \cdot S_{old}(q)$$

$$D_{new}(p) = \begin{cases} \dfrac{1}{N(p)} \displaystyle\sum_{q \in Neigh(p)} w(q,p) \cdot S_{old}(q) \cdot D_{old}(q) & \text{for } N(p) > 0 \\ 0 & \text{for } N(p) = 0 \end{cases}$$

$$S_{new}(p) = \begin{cases} 1 & \text{for } N(p) > 0 \\ 0 & \text{for } N(p) = 0 \end{cases}$$

   where $Neigh(p)$ denotes a set of spatial neighbours of pixel p and $w(q,p) \geq 0$ denotes a weight for neighbour q of pixel p.
4. Set $S_{old} = S_{new}$ and $D_{old} = D_{new}$.

5. Repeat step 2, 3 and 4 until S_{new} consists completely of ones.
6. Set the output motion field to $D_E = D_{new}$.

The algorithm starts with a "processed" area containing valid motion, defined by a given shape field (i.e. $S_{r,e}$). For each pass trough the field, all pixels that are adjacent to the processed area are set to an average of those neighbouring pixels that lie inside the processed area. As a final result thereof the whole area of the resulting motion field contains valid motion information. In the case. of using a backward motion compensation scheme D_E is identical with D, i.e. $D_E = D$.

In the next step the areas in the reference image which must be extended are determined. These areas are defined by the shape $S_{r,\mu}$ and can be calculated by the following steps:

1. Extending the shape $S_{r,e}$ using the motion field and the shape S_u resulting in the shape $S_{r,E}$. For example: warping the shape S_u according to the motion field from the position of I_n to the reference position resulting in $S'_u = \text{OppositeWarp}(S_u, D_E)$ and setting $S_{r,E} = S_{r,e} \vee S'_u$. Holes within $S_{r,E}$ may be filled.
2. Calculation of $S_{r,\mu}$: $S_{r,\mu} = S_{r,E} \wedge \neg S_{r,e}$.

In the next step the extended reference image $I_{r,E}$ 1150 can be formed by warping of those areas of I_n which are defined by S_u to those of $I_{r,e}$ which are defined by $S_{r,\mu}$ according to the motion field D_E, e.g., $I'_n = \text{OppositeWarp}(I_n, D_E)$; $S'_n = \text{OppositeWarp}(S_n, D_E)$; $I_{r,E} = I_{r,e} + I'_n$ under the consideration of S'_n and $S_{r,\mu}$. If necessary an extrapolation process is added in order to fill all areas defined by $S_{r,\mu}$.

The added pixels that were not used for the reconstruction of I_n can now be removed, resulting in the final version of the extended reference image 1160. This can be done by finding the smallest rectangle containing the shape field $S_{r,E}$. The difference vector between the upper left corners of $I_{r,E}$ and the found smallest rectangle must be removed from the motion field in the same manner as described above in situations where forward motion compensation is employed. All subspace models which are defined in reference position must be extended, too.

Tenth preferred embodiment

There may be cases where a sequence is broken into multiple sets of consecutive images, though some similarity between the images remains between the sets of consecutive images. One such similarity is when the reference image of a first set of consecutive images is similar to the reference image of a second set of consecutive images. The second reference image, called a P-image, can then be reconstructed from a prediction based on the first reference image plus a residual.

One method is to first encode each set of consecutive images independently. Then, as part of compressing the resulting model, the reference image of the second set of consecutive images can be encoded as a predicted version of the first one. Therefore a motion estimation between the two reference images can be performed and the motion field plus a residual is transmitted.

Another method is to let the motion between the first and the second reference image be included in the subspace model of motion for the first set of consecutive images.

A possible resulting structure is illustrated in fig. 12. A first reference image 1210 is used as a basis for motion estimation to several images 1212, 1214, 1216, 1218. The motion fields 1240 can be modelled together as a subspace model. One image 1218 is then used as a reference image for motion estimation to further images 1220, 1222, 1224, 1226, 1228. One image 1228 of these can once more be used as a reference for motion estimation to further images 1230, 1232, 1234.

A related technique without using a subspace model of motion, is used in the MPEG class of encoders.

For the compression of the residuals pertaining to P-images "inter" tools may be used instead of "intra" tools.

Eleventh preferred embodiment

There are images that cannot be predicted sufficiently with only one IDLE model, even comprising an extended reference image. Consider figs. 13 a - c. A first image 1300 in a unit of images shows a woman 1305 partly occluded by a car 1310, together with a man 1315. A later image 1360 shows the woman 1365 fully, but this time the man 1375 is partly occluded by the car 1370. An intermediate image 1330 shows the woman 1335 and the man 1345 partly occluded by the car 1340. For this very short sequence it is not possible to build a set of consecutive images and select one reference image for a single IDLE model that can be used to predict both the woman 1305, 1335, 1365 and the man 1315, 1345, 1375 fully satisfactory.

Assuming that the two images 1300 and 1360 were the first and last image in a unit of images, it is possible to predict each part of the intermediate image 1330 from either the first 1300 or the last 1360 image.

This leads to the principle of combining forward and backward predictions. One early (e.g. the first) and one late (e.g. the last) image of a unit of images are selected as reference images for two IDLE models, such that their sets of consecutive images (SCIs) overlap, that is both contain a common subset of the unit of images. Each non reference image within the intersection can therefore be predicted by the first, early IDLE model, i.e. by a forward prediction, and by the second, later IDLE model, i.e. by a backward prediction. In the simplest case for each spatial block of pixels or for each pixel in such an image, it is measured, whether the forward prediction or the backward prediction produces the better prediction. The encoder sends one bit for each spatial block of pixels or pixel telling the decoder whether to use the forward or the backward prediction.

The resulting structure is illustrated in fig. 14. An IDLE model 1440 with reference image 1412 is used for predicting nearby images 1410, 1411, 1413, 1414, 1415, 1416. Another IDLE model 1445 with reference image 1416 also covers nearby images 1414, 1415, 1417, 1418, 1419, 1420. Those images 1414, 1415 that are covered by both models are called bi-directional predicted images, or B-images in short. Those images 1410, 1411, 1413 that are only covered by one model are called unidirectional predicted images, or U-images in short. The structure can be continued, in that a further IDLE model 1450 with reference image 1421 covers nearby images 1417, 1418, 1419, 1420, 1422, 1423, 1424 and1425, etc.

Whether an image is encoded as a U-image or a B-image depends on the quality with which it can be predicted from a single IDLE model. The decision may be done while building the IDLE models.

A B-image can change to a U-image within the decoder if random access makes the earlier of the two models unavailable. For example the decoder gets first access to the sequence at IDLE model 1450 and has therefore no information from the previous IDLE model 1445. Hence the images 1417, 1418, 1419 and 1420 may be decoded as U-images.

In order to tell the decoder how to build a B-image, a blend field is calculated in the encoder and transmitted to the decoder. Each element of the blend field contains one value for each pixel or each spatial block of pixels of the image, indicating whether it has to be taken from the forward prediction of the earlier IDLE model or the backward prediction of the later IDLE model. For

example the value 0 may indicate the usage of the forward prediction and the value 1 the usage of the backward prediction.

The blend fields will often have spatial redundancy. If one pixel or one block of pixels in a B-image is taken from the forward prediction, then there is an increased probability that the neighbouring pixel or block of pixels will also be taken from the forward prediction. Each blend field can therefore be compressed using a compression tool for binary images. Due to the binary character of the blend fields all methods for coding shape fields can also be used for blend fields.

The blend fields will also often have temporal redundancy. If one pixel or one block of pixels in a B-image is taken from the backward prediction, then there is an increased probability that the corresponding pixel or block of pixels in the following B-image will be taken from the backward prediction, too.

The temporal and spatial redundancy of the blend fields occurs for areas of the image where the prediction is good in one direction and less good in the other. In areas of the image where both the forward and backward prediction produce equally good quality or fidelity the selection of the best may be influenced by secondary effects, like noise in the images. The alternation between forward and backward predictions might produce visual artefacts in the B-image. The blend fields will tend to be less compressible. It may be advantageous to apply techniques which simplify the blend fields in those areas where the forward and backward predictions are of similar quality.

One such technique is to apply a median filter to the blend fields in those areas where the forward and backward predictions have similar fidelity or quality.

Another technique is to use methods which adapt the blend value for each element according to a compromise between prediction quality and similarity to neighbours. One such method, called Agree filter, minimises a cost function by performing the following steps:

(1) Calculate for each element in the blend field the value that minimises the prediction error of the corresponding B-image.
(2) Calculate for each element a value in a new blend field that minimises the cost function which depends on the prediction error of the B-image and on the dissimilarity of the value with the neighbour blend field values.
(3) Set the blend field equal to the new blend field.
(4) Repeat steps (2) and (3) for a certain number of times, e.g. until the process converges.

Performing above steps the Agree filter determines for each element p of the blend field B a value b_0, i.e. $B(p) = b_0$, which minimises the cost function $F_p(b)$, i.e. $F_p(b) \le F_p(b_0) \quad \forall b \in \{0,1\}$. Examples for the function $F_p(b)$ are:

1. $F_p(b) = \sum_{q \in Block(p)} |(b-1)I_1'(q) + bI_2'(q) - I(q)|$

2. $F_p(b) = \sum_{q \in Block(p)} |(b-1)I_1'(q) + bI_2'(q) - I(q)| + k \sum_{n \in Neigh(p)} \sum_{q \in Block(n)} |(b-1)I_1'(q) + bI_2'(q) - I(q)|$

3. $F_p(b) = \sum_{q \in Block(p)} |(b-1)I_1'(q) + bI_2'(q) - I(q)| + k \sum_{n \in Neigh(p)} |B(n) - b|$

4. $F_p(b) = \left(1 + \sum_{q \in Block(p)} |(b-1)I_1'(q) + bI_2'(q) - I(q)|\right)\left(1 + k \sum_{n \in Neigh(p)} |B(n) - b|\right)$

where I_1' denotes the forward prediction of the first IDLE model, I_2' the backward prediction of the second IDLE model, I the original image to be reconstructed, B the blend field, k a weighting constant, $Block(p)$ a spatial block of pixels in the image corresponding to the element p in the blend field and $Neigh(p)$ a set of spatial neighbour elements of element p in B. For the first two examples no iteration (i.e. repetition of step (2) and (3)) must be performed. In the case of the last two examples the second term measures how many of the neighbouring blocks have a blend value that is different from the current one. The constant k determines the strength of influence of the second quantity.

The above technique, which was shown for block-wise decisions on whether to use the forward or backward prediction model in the predicted images, can also be used with block size one, that is, for each pixel. Then the conditioning of the blend fields, e.g. as described for the Agree filter, and compression of the same fields, become more important.

The blend field can be seen as a weight field. Instead of using only either 0 or 1 weights, the weights can be feathered. Consider a transition from backward to forward prediction in the middle of eight pixels on a scan line. Without feathering, the weights for the forward prediction may be

0 0 0 1 1 1,

while the weights for the backward predicting are

1 1 1 0 0 0 .

With feathering, this may become, for the forward prediction

0 0 0.25 0.75 1 1,

and for the backward prediction:

1 1 0.75 0.25 0 0.

The resolution of the weights in the blend fields B can be increased until real valued numbers between 0 and 1 are reached. blend fields with real valued weights in the unit interval, which are well compressible may be generated by a procedure as discussed in "Compactly-Encoded Optical Flow Fields For Bidirectional Prediction and Video Coding", Ravi Krishnamurthy, Pierre Moulin, John W, Woods, Visual Communications and Image Processing, San Jose, California, 1997, which is hereby included by reference. A similar concrete realisation of such a procedure is described in the following:

For each pixel or block of pixels the final prediction I' is a linear combination between the forward prediction I_1' and the backward prediction I_2' with a corresponding value of the blend field B:

$$I' = (1-B)I_1' + BI_2' \quad \text{with} \quad B \in [0,1]$$

For each pixel or block of pixels the prediction error E is then given by

$$E = I - I' = E_1 + B(E_2 - E_1)$$

where I is the original image and $E_i = I - I_i'$ are the prediction errors (residuals) for both predictions I_i'. In order to achieve $E = 0$ (vanishing prediction error) the blend field B would have to be

$$B = \frac{E_1}{E_1 - E_2} \quad \text{for} \quad E_1 \neq E_2 .$$

But these values are not restricted to the interval $[0,1]$. Moreover such a blend field might be rough and therefore hard to compress. A better compressible blend field with values between 0 and 1, which usually improves prediction quality can be found by solving the following variation problem with respect to two constraint functions. The first constraint is given by

$$F_1 = \chi_{\{0\}}(E_2 - E_1)\left(B - g\left(\frac{E_1}{E_1 - E_2}\right)\right)^2,$$

with the indicator function

$$\chi_M(x) := \begin{cases} 1 & x \in M \\ 0 & x \notin M \end{cases}$$

and the "ramp function"

$$g(x) := x\chi_{[0,1]}(x) + \chi_{]1,\infty[}(x).$$

It represents the deviation from a blend field with values restricted to the interval $[0,1]$, which favours vanishing prediction error. The second constraint

$$F_2 = \left(\frac{\partial B}{\partial v}\right)^2 + \left(\frac{\partial B}{\partial h}\right)^2 = B_v^2 + B_h^2$$

represents the deviation from a smooth blend field. The quantities v and h are the co-ordinates of the elements of B. The derivatives $\partial/\partial v$ and $\partial/\partial h$ indicate, that a continuous representation of the problem has been chosen in order to use the calculus of variation. The resulting differential equation will be rewritten in discrete form later on. The two constraints are combined to

$$F = F_1 + \tilde{\alpha}^2 F_2$$

where the relative weight of the smoothness term is controlled by the parameter $\tilde{\alpha}$. The wanted blend field minimises the functional

$$\Psi = \iint F(B, B_v, B_h, v, h)\, dvdh$$

where the integration covers the whole image. The minimisation of Ψ with respect to B can be performed using the calculus of variations. The corresponding Euler-Lagrange equation leads to the linear partial differential equation

$$\chi_{\{0\}}(E_2 - E_1)\left(B - g\left(\frac{E_1}{E_1 - E_2}\right)\right) = \tilde{\alpha}^2 \Delta B$$

which can be solved by standard methods as described in "Numerik partieller Differentialgleichungen ", Ch. Großmann, H.-G. Roos, Teubner, Stuttgart 1994.

E.g. the above equation can be transformed into a system of linear algebraic equations by replacing the Laplace operator Δ by a discrete approximation

$$\Delta B(v,h) = \left(\frac{\partial^2}{\partial v^2} + \frac{\partial^2}{\partial h^2}\right) B(v,h) \approx 3\left(\bar{B}(v,h) - B(v,h)\right)$$

with

$$\bar{B}(v,h) = \frac{1}{6}\left[B(v-1,h) + B(v+1,h) + B(v,h+1) + B(v,h-1)\right] +$$
$$+ \frac{1}{12}\left[B(v-1,h-1) + B(v-1,h+1) + B(v+1,h-1) + B(v+1,h+1)\right].$$

Setting $\alpha^2 = 3\tilde{\alpha}^2$, using above approximation for the Laplace operator Δ and introducing the weight

$$w(\alpha, E_1 - E_2) := \frac{\alpha^2}{\alpha^2 + \chi_{\{0\}}(E_1 - E_2)}$$

leads to the system of equations

$$B = w(\alpha, E_1 - E_2)\bar{B} + \left(1 - w(\alpha, E_1 - E_2)\right) g\left(\frac{E_1}{E_1 - E_2}\right).$$

Above equations can be solved iteratively, e.g. by a successive over relaxation algorithm. Such an algorithm is described in "Numerische lineare Algebra", W. Bunse, A. Bunse-Gerstner, Teubner Studienbücher, Stuttgart 1985, which is hereby included by reference.

In order to avoid the influence of artefacts, which are not visible by the human visual system, the prediction errors (residuals) E_i may be pre-processed by methods described in "Method and apparatus for compression of video images and image residuals", already included by reference. Residuals corresponding to artefacts below a visual threshold may be removed from the

prediction error E_i before the calculation of the blend field starts. This can be done using the modules DefBounds 866 and RelToBounds 867 described in the fifth preferred embodiment.

In order to reduce the amount of data required for the blend fields, the set of blend fields of successive images can be compressed by building a subspace model with reduced rank, as described for the motion data (module BuildSSMod 190). Each basis image of the subspace model has the size of a blend field. Each factor has one temporal weight coefficient for each B-image covered by the subspace model. On the decoder side, the blend field for one bi-directional described image (B-image) is found by multiplying the temporal weight coefficients corresponding to the image with the basis images.

The subspace model may be build in one step using stored blend fields or may be updated recursively (as done for motion fields in module UpdateSSMod 540, see third preferred embodiment). An intermediate determination of the rank and a final determination may be combined (see seventh preferred embodiment). All blend fields between two reference images may be described by one subspace model.

The blend fields may be warped to a common position within the sequence before modelling. One way of doing this is to calculate the blend fields in the corresponding image positions, as described above. Then each blend field may be warped to the position corresponding to the reference image of e.g. the later IDLE model. This can be done by using the OppositeWarp routine with the motion field obtained from the later IDLE model. Then a subspace model of these blend fields is build. On the decoder side each blend field resulting from the blend field subspace model is warped to its corresponding image position with the corresponding motion field obtained from the subspace model of motion using the Warp routine. This warped field is used to perform the blending of the forward and backward predictions.

Before transmitting the subspace model for the blend fields from the encoder to the decoder it can be compressed. The basis images of the model can be compressed using standard image compression techniques, like entropy coding, run-length coding, predictive coding, DCT or similar. In particular, they can be compressed using the same or similar routines as for the subspace models of motion, as described in the second preferred embodiment. Moreover the basis images may be sub-sampled. Predictive coding techniques and frequency transforms may be applied for the compression of the temporal weight coefficients.

Blend fields with values exclusively 0 and 1 can be compressed by applying techniques for shape fields or shape residuals as described in the ninth embodiment.

An example for a realisation of an encoder which works with subspace models for blend fields containing values in the unit interval will now be described with reference to fig. 15a. The encoder Enc 1500 receives as input a sequence of images 1501 which are buffered in the module ImageStore 1502. The module BuildModel 1503 builds an IDLE model which is compressed in module CompMod 1504. This compressed model is simultaneously put to the module DecompMod 1505 and to the module TraMod 1509 where it is transmitted. The module DecompMod 1505 decompresses the IDLE model. The decompressed model is simultaneously stored in the module ModelBuffer 1506 and put to the module CalcBlendModel 1507. The module CalcBlendModel 1507 receives the previous IDLE model stored in the module ModelBuffer 1506, the current IDLE model from the module DecompMod 1505 and original images from the module ImageStore 1502. The module CalcBlendModel 1507 calculates the corresponding blend subspace model which is then compressed in the module CompBlendMod 1508. Finally the module TraBlendMod 1510 transmits the compressed blend subspace model.
The module CalcBlendModel 1507 will now be described with reference to fig. 15b. The module 1520 receives a earlier 1521 and a later IDLE model 1522 and original images 1523. The module BlendControl 1524 calculates the intersection of the SCIs of the two IDLE models and uses this information in order to control the modules Predict 1525 and 1526. These modules build predictions (see module Predict 320 in fig. 3) for all images lying in the intersection of the SCIs. The module CalcBlendFields 1527 receives these predictions as well as the original images and calculates blend fields for all predictions. The module BuildSSModBlend 1528 builds a subspace model of these blend fields.

The module CalcBlendFields 1527 will now be described with respect to fig. 15c. The module 1530 receives a forward prediction 1531 from an earlier IDLE model, a backward prediction 1533 from a later IDLE model and the corresponding original image 1532. The modules CalcResidual 1534, 1535 calculate the residuals (prediction errors) taking into account the predictions and the original image. The modules CalcResidual 1534, 1535 may also process the residuals by methods described in "Method and apparatus for compression of video images and image residuals" already included by reference. Using the two residuals the module CalcValues 1536 calculates all values for the output blend field by solving the variation problem described above.

An example for a corresponding decoder will now be described with reference to fig. 16. The decoder Dec 1600 receives an IDLE model 1601 in the module ReceiveMod 1610 and a blend subspace model 1602 in the module ReceiveBlendMod 1620. The module DecompMod 1615 decompresses the IDLE model and the module DecompBlendMod 1616 decompresses the blend subspace model. The decompressed blend subspace model is put to module RebuildBlendFields 1650. The decompressed IDLE model is simultaneously stored in the module ModelBuffer 1630

and put to the modules BlendControl 1640 and Predict 1670. The module BlendControl 1640 calculates the intersection of the SCIs of the current IDLE model and the previous IDLE model received from the module ModelBuffer 1630 as well as the unit of images between the reference images of both models. The module Predict 1660 calculates all forward predictions of the previous IDLE model. The module Predict 1670 calculates all backward predictions of the current IDLE model. Both modules 1660, 1670 receive control information from the module BlendControl 1640. If a forward and a backward prediction exist for a certain image, the module BlendPredictions 1680 blends them according to their corresponding blend field received from the module RebuildBlendFields 1650. If only one prediction exists for a certain image, this prediction passes the module BlendPredictions 1680 unaltered. The described process is controlled by information received from the module BlendControl 1640.

Twelfth preferred embodiment

There may be cases where an image cannot be predicted well from a reference image using motion information. Some special cases could be solved, like occlusions and innovations, as described in the previous preferred embodiment, or objects that enter or leave the scene during a set of consecutive images, as described using extended reference images. But many cases, including cases of unsystematic motion or intensity changes, still cannot be described efficiently by an IDLE model. These areas may have visible artefacts, which in many applications are not tolerable. Such areas are called model failure areas.

In order to correct model failure areas, residuals as the differences between original images and predictions can then be transmitted. Optionally, the original area instead of the difference can be transmitted if it needs less bits.

This is illustrated in fig. 17, which shows an encoder Enc 1700. A sequence 1701 of images is kept in an image store 1705. BuildMod 1710 reads an SCI from the image store 1705 and builds an IDLE model. The model is compressed in the module CompMod 1722 and then transmitted by TraMod 1725. The model is also used for predicting the images in a module Predict 1715 (see module 320 in fig.3). A module 1720 calculates the differences between the original images and the predictions, producing residuals. These are compressed in module CompRes 1723 and then transmitted by TraRes 1730. In this scenario, the residuals are calculated with the predictions obtained by the unprocessed model.

Alternatively, quantisation or other compression artefacts coming from lossy compression of the model can be considered when computing the residuals. This can be done by calculating the residuals with the predictions obtained by the processed model, which has passed compression

and decompression. A corresponding encoder is illustrated in fig. 18. The encoder Enc 1800, receives a sequence of images 1801 and stores it in the ImageStore 1810. BuildMod 1820 reads an SCI from the ImageStore 1810 and builds an IDLE model. The IDLE model is compressed in module CompMod 1830, where quantisation, thresholding or other types of lossy data reduction take place. The compressed model is transmitted by TraMod 1880 and put to module DecompMod 1840, where it is decompressed. This decompressed model is used for predicting the images in module Predict 1850. The residuals as the differences between the original images and these predictions are calculated in module 1860. The residuals are compressed in module CompRes 1870 and then transmitted by TraRes 1890.

All residuals can be compressed as described for the residuals pertaining to P-images. For residuals not pertaining to P-images parameter sets leading to worse quality may be used. Assuming that the relevant parts of such residuals are located in a relatively small area, direct addressing and coding only the blocks containing these relevant parts may yield better results. Predictive coding techniques in combination with variable length coding may be used for a loss-less compression of these addresses. The set of addresses can be interpreted as a binary field and can therefore be compressed with tools for bi-level images.

Different parts of an image may have different tolerances regarding how large residuals are allowed at a given level of reconstruction quality, due to masking effects in the human visual system. Some of those masking effects together with ways of exploiting them, are described in the patent application "Method and apparatus for compressing image residuals", already included by reference. A module that processes (or dampens) the residuals according the rules described in this patent application can be inserted before module CompRes 1870.

To achieve the highest possible fidelity, the residuals should be computed on the encoder side based on a full simulation of the decoder. That is, the encoder should incorporate all artefacts of compression of the reference images and the subspace model of motion. This way it is guaranteed that transmitting the residuals will indeed improve the fidelity of the reconstructed images.

When the reference image is compressed with a lossy technique, it will exhibit coding artefacts. Those artefacts will then manifest themselves also in predictions, which implies that one compression artefact may have to be corrected in residuals for several images. This will degrade the compression ratios.

In order to avoid unnecessary coding of quantisation artefacts, the residual can be computed as if the reference image had been compressed in a loss-less way. That is, for purposes of computing residuals, the original reference image is used instead.

A similar argument can be applied for motion data. Therefore, the residual may be computed as if the basis images had been compressed in a loss-less way.

A similar argument also applies regarding the rank of the subspace model of motion. The encoder chooses a number of factors as a compromise between bit rate and quality. Some factors that are not transmitted describe motion that is visually insignificant, but whose removal still leads to an increase in residuals. It may be beneficial to compute the residuals as if a higher number of factors has been used for the subspace model of motion.

Characteristics of the current motion field and the subspace model of motion may be considered by estimating the visibility of artefacts caused by the rebuilt motion. E.g., a spatially and temporally smooth error in motion, with a magnitude of e.g. 1 pixel, may be hardly visible in a larger area with a smooth motion field. A motion artefact with a magnitude of 0.5 pixels might be easily visible in an area with a non-smooth motion field. Such information can be exploited when deciding whether original or quantised motion should be used for determination of predictions.

Thirteenth preferred embodiment

The P-images defined in the seventh embodiment are reconstructed by adding a residual image to the prediction of an IDLE model containing an earlier reference image. This residual image can in principle be treated similar to the residuals occurring in the previous embodiment that is visually insignificant residuals may be suppressed or dampened. Principles according to "Method and apparatus for compression of video images and image residuals", already included by reference, may therefore be used for dampening the residuals, so that only visually significant residuals need to be compressed.

One fact influencing the visibility of image artefacts is the temporal extent to which such artefacts are present. One method for limiting the temporal extent is to switch between dampening residuals and not dampening residuals for P-images.

This is illustrated in fig. 19, showing an encoder Enc 1900. A sequence 1901 of images is buffered in an image store 1910. BuildMod 1920 reads a set of consecutive images from the image store 1910 and builds a corresponding IDLE model. CompMod 1930 compresses this

model, which is then transmitted in TraMod 1980.The same model is decompressed in module DecompMod 1935. The module Predict 1940 receives the decompressed model and predicts the images (see module 320). The difference module 1950 calculates the difference between the original image and its prediction, producing a residual. A switch 1960, controlled by the encoder control module EncCont 1905 directs the residual either to the module DampRes 1970 for dampening, or to the module CompRes 1975. The output of module DampRes 1970 is also put to module CompRes 1975, which compresses the residuals. The compressed residual is transmitted by module TraRes 1990.

The switching can be done in a way such that residuals are transmitted without being dampened for a certain fraction of the P-images, e.g. every second P-image.

Alternatively, the switching can be done with respect to time, expressed in the number of images or in seconds. E.g., the switching can be done so that if more than one second has elapsed since the last I-image or not-dampened P-image residual was transmitted, then dampening should be disabled for the next P-image residual, else it should be enabled.

Alternatively, the encoder can control the parameters of DampRes instead of operating a switch. For the case of the method given in "Method and apparatus for compression of video images and image residuals", already included by reference, a bit rate control parameter affecting bounds for allowed residuals can be used. The parameters can be set so that P-image residuals are normally transmitted with strong dampening, while some chosen P-image residuals are transmitted without or with very weak dampening.

This may be extended to several levels, e.g. so that P-image residuals are normally transmitted with strong dampening, some are transmitted with weaker dampening, and some are transmitted without dampening.

The dampening of a P-image residual may be performed for certain pixels or certain blocks of pixels only. These pixels or blocks of pixels may be selected randomly, such that a certain number of pixels or blocks of pixels are dampened. This number may be a function of the time since the last I-image was transmitted.

Fourteenth preferred embodiment

In addition to systematic changes in motion, an input sequence may also exhibit systematic changes in intensity. Those intensity changes may also be modelled efficiently using subspace

models. The basic method is explained in the patent application "Method and Apparatus for data analysis", already included by reference.

Subspace modelling of intensity can be incorporated in the present invention, as shown in fig. 20. An encoder Enc 2000 receives as input a sequence of images 2001. For a set of consecutive images SCI read from the image store 2030, a subspace model of motion is built in BuildModMotion 2040. This subspace model of motion, together with the reference image and the original images are used to build a subspace model of intensity changes in BuildModInt 2050. The total output is compressed in module CompMod 2055 and transmitted in module TraMod 2060.

In the following two possible methods to employ a subspace model of intensity changes in order to improve the prediction process are described.

In the first method an intensity change $\Delta I'_n$ is added to the reference image I_{ref} before warping with respect to the rebuilt motion field D'_n is performed, which is expressed as

$$I'_n = \text{Warp}(I_{ref} + \Delta I'_n, D'_n).$$

The intensity change $\Delta I'_n$ is rebuilt from the corresponding subspace model of intensity changes. Such a subspace model can be built in two ways described in the two alternative versions of the module BuildModInt, which are shown in figures 21a and 21b, respectively.

One possible version of module BuildModInt will now be described with reference to fig. 21a. A set of consecutive images SCI 2101 and a corresponding IDLE model consisting of a motion model MotionSSMod 2102 and a reference image I_{ref} 2103 are given as input. RebuildMotion 2110 rebuilds each motion field D'_n from the subspace model MotionSSMod 2102. In module OppositeWarp 2120 each image I_n of the SCI 2101 is warped to the reference position using the corresponding motion field. The difference ΔI_n between this image and the reference image is calculated by module 2130. Hence ΔI_n is given by

$$\Delta I_n = \text{OppositeWarp}(I_n, D'_n) - I_{ref}$$

One such difference image for each image in the set of consecutive images is the input to BuildSSMod 2140 which delivers as output 2141 the subspace model of intensity changes.

An alternative version of the module BuildModInt will now be described with reference to fig. 21b. The same input as in fig. 21a is given. For each image I_n of the SCI 2151 the reference image I_{ref} 2153 is warped in Warp 2160 according to the corresponding motion field D'_n which was rebuilt from the motion subspace model 2152 by RebuildMotion 2155. The warped reference image is therefore a prediction for the corresponding original image I_n. The difference between I_n and this prediction is computed in 2170. This difference is warped in OppositeWarp 2180 to the reference position according to D'_n. Hence the final warped difference ΔI_n is given by:

$$\Delta I_n = \text{OppositeWarp}\left(I_n - \text{Warp}\left(I_{ref}, D'_n\right), D'_n\right)$$

One such warped difference for each image in the set of consecutive images is the input to BuildSSMod 2190.

In order to handle such a subspace model of intensity changes in a decoder (fig.3) the module Predict 320 must be altered. Such a version of module Predict will now be described with reference to fig. 22a. The module 2200 receives an IDLE model 2201 and sends the reference image to the module RefImgSto 2202. The subspace model of intensity changes is sent to the module RebuildIntensChange 2203 and the subspace model of motion is sent to the module RebuildMotion 2204. For each image within the corresponding SCI of the IDLE model the rebuilt intensity change from 2203 is added to the reference image in module 2205. The module Warp 2206 warps the resulting images according to the corresponding rebuilt motion fields from 2204 and delivers predictions 2207 for all images of the SCI. The reference image itself passes the modules 2205 and 2206 unchanged.

In general the performance of IDLE models which contain above described subspace model of intensity changes differ in all earlier embodiments and therefore the pertaining Predict modules 480, 1525, 1526, 1660, 1670, 1715, 1850 and 1940 must work like 2200.

In a second possible method to employ a subspace model for intensity changes, the rebuilt intensity change $\Delta I'_n$ is added to the warped reference image, which is expressed as

$$I'_n = \text{Warp}(I_{ref}, D'_n) + \Delta I'_n .$$

Again the corresponding subspace model of intensity changes can be built in two ways.

In the first way for each image I_n of the SCI the reference image is warped according to the corresponding rebuilt motion field D'_n to get a prediction of I_n. Then the difference ΔI_n between I_n and this prediction is calculated. Hence ΔI_n is given by

$$\Delta I_n = I_n - \text{Warp}(I_{ref}, D'_n)$$

One such difference image for each image in the SCI is used to build the subspace model of intensity changes.

In the second way each image I_n of the SCI is warped to the reference position according to the corresponding rebuilt motion field D'_n. The difference between this image and the reference image is then warped to the position of I_n according to D'_n. Hence the final warped difference ΔI_n is given by;

$$\Delta I_n = \text{Warp}\left(\text{OppositeWarp}(I_n, D'_n) - I_{ref}, D'_n\right)$$

Again one such difference image for each image in the SCI is used to build the subspace model of intensity changes.

In order to handle the subspace models of intensity changes employed according to the second described method the module Predict 320 of the decoder (fig.3) must work like module Predict 2220 which will now be described with reference to fig. 22b. The module 2220 receives an IDLE model 2221 and sends the reference image to the module RefImgSto 2222. The subspace model of intensity changes is sent to the module RebuildIntensChange 2224 and the subspace model of motion is sent to the module RebuildMotion 2223. For each image within the corresponding SCI of the IDLE model the module Warp 2225 warps the reference image according to the corresponding rebuilt motion field from 2223. The module 2226 adds the corresponding rebuilt intensity changes from 2224 to the warped reference images and delivers predictions 2227 for all images of the SCI. The reference image itself passes the modules 2225 and 2226 unchanged.

In general the performance of IDLE models which contain a subspace model of intensity changes employed according to the second described method differ in all earlier embodiments and therefore the pertaining Predict modules 480, 1525, 1526, 1660, 1670, 1715, 1850 and 1940 must work like 2220.

Modelling intensity changes before warping can have advantages regarding compression performance when moving objects change intensity systematically. Modelling intensity changes after warping can have advantages regarding implementation simplicity.

Some types of residuals are visually more disturbing than others. This may be considered when modelling intensity changes. E.g., if a sharp intensity edge is positioned one quarter of a pixel out of position in a predicted image, this will lead to a large intensity residual, though the artefact may not be visible. Further, it is a difficult problem to estimate the motion fields for moving objects at the boundary of objects, and the residuals may even be unsystematic in magnitude. Since such residuals are large, they may influence the subspace modelling. Since they are unsystematic, they may lead to a high number of factors necessary to obtain good fidelity. Since they occur at edges, they may sometimes be visually insignificant. It is therefore advantageous to let such pixels have less influence on the subspace modelling of intensity changes.

One method is to pre-process the intensity residual image before modelling. This is illustrated in fig. 23, showing an alternative BuildModInt 2100. A reference image 2303, a subspace model of motion MotionSSMod 2302 and a corresponding set of consecutive images SCI 2301 are given. Motion fields are reconstructed in RebuildMotion 2310, every image is warped to reference position in OppositeWarp 2320, and the difference 2330 between this image and the reference image is fed to Damp 2340 where the residual is dampened, considering masking effects derived from the reference image as described in "Method and apparatus for compression of video images and image residuals", already included by reference. The dampened differences are then used for building a subspace model in BuildSSMod 2350. The same procedure can be applied for all variations to build a subspace model of intensity changes.

Another method is to use weights in the subspace modelling. Instead of removing intensity differences of relatively little visual importance from the residual before subspace modelling, the intensity differences can be allowed to remain, and the weights in the subspace modelling are changed instead. Residuals of little visual importance can then be given low weights. For this purpose, BuildSSMod 2350 must be able to handle pixel by pixel weights for each image.

As described above, the modelling of intensity depends on the rebuilt motion field, represented by D_n'. Three kinds of rebuilt motion fields are possible, which differ in the kind of processing they had been exposed to.

A first possibility is to use each motion field directly as calculated by the motion estimation tool.

A second possibility is to rebuild the motion field from the subspace model of motion after determining the rank of the subspace model.

A third possibility is to rebuild the motion field from the subspace model of motion, after the subspace model has been compressed and decompressed.

The three possibilities have an increasing ability to produce a model that will lead to high fidelity in the final decoded image, in that the two last possibilities compensate for the effects of motion subspace modelling and the third possibility further compensates for the effects of compression and decompression of the motion subspace model.

This increased fidelity comes at the expense of increased bit rate, since the motion artefacts introduced by subspace modelling and compression may lead to intensity change patterns that do not always fit well into a subspace model of intensity changes.

Fifteenth preferred embodiment

In some cases, IDLE models corresponding to consecutive SCIs may still exhibit a high amount of correlation. One way of exploiting this for the reference image was shown in the seventh preferred embodiment. A similar effect can be exploited for basis images of the subspace model of motion or basis images for the subspace of intensity changes for consecutive SCIs, that is, the pertaining basis images span similar vector spaces. Let $I_f(k)$ denote the row vector representing the f -th basis image of a subspace model for a certain attribute (motion, intensity changes, etc.) belonging to the k -th IDLE-model. Let $I_B(k) = \left[I_1(k)^T \quad I_2(k)^T \quad \ldots \quad I_F(k)^T \right]^T$ represent the matrix containing F row vectors describing a basis B of the subspace model for the certain attribute belonging to the k -th IDLE-model. The set of current basis images $I_B(k)$ may be represented as the sum of the preceding basis images $I_B(k-1)$ and a correction term $R_B(k)$, which can be regarded as a residual, that is,

$$I_B(k) = I_B(k-1) + R_B(k).$$

The residual $R_B(k)$ may be transmitted to the decoder instead of the full basis images $I_B(k)$. This may bring a gain in terms of data compression if the representation of the residual $R_B(k)$ requires less data than the representation of the current basis images $I_B(k)$.

Although the subspaces spanned by the IDLE models corresponding to two consecutive SCIs may be very similar, the pertaining basis images may still differ substantially. To match the basis images an invertible matrix $M(k)$ may be determined to achieve

$$I_B(k) = M(k) \cdot I_B(k-1) + R_{B,M}(k),$$

such that the basis images $I_B(k-1)$ for the previous subspace model are transformed to coincide as well as possible with the basis images $I_B(k)$ of the current subspace model, where the matrix $R_{B,M}(k)$ accounts for the mismatch between the basis images (note: $R_B(k) = R_{B,M}(k)$ if $M(k)$ is the identity matrix).

The purpose of the procedure described in the following is to provide a matrix $M(k)$ and a corresponding matrix $R_{B,M}(k)$ so that $I_B(k)$ can be constructed in the decoder. For the purpose of data compression, the representation of the matrices $M(k)$ and $R_{B,M}(k)$ should require less bits as the representation of the current basis images $I_B(k)$. To this end, the matrix $M(k)$ should be computed as to minimise the amount of data required for the representation of $R_{B,M}(k)$ without sacrificing fidelity in the prediction of $I_B(k)$. One method to compute such a matrix $M(k)$ is to minimise the sum of squares of $R_{B,M}(k)$, e.g. by using a standard least squares estimation technique as given by

$$M(k) = I_B(k) \cdot I_B(k-1)^T \cdot \left[I_B(k-1) \cdot I_B(k-1)^T \right]^{-1}.$$

A final decision may be made on the transmission of either the current basis images $I_B(k)$ directly or the matrix $M(k)$ together with the residual $R_{B,M}(k)$, whatever needs fewer data. This decision can also be performed row by row, i.e. for each factor.

The implementation structure for this preferred embodiment is illustrated in Fig. 24a. The encoder Enc 2400 receives images 2401 which are buffered in the module ImageStore 2402. BuildMod 2404 reads images from the module ImageStore 2402 and builds an IDLE model for a set of consecutive images (SCI). The resulting IDLE model is simultaneously passed to the module CalcModRes 2410, to the module ModelBuffer 2406, to the module Predict 2408 and to the module CompMod 2418. The module CalcModRes 2410 receives the current IDLE model from BuildMod 2404 and the preceding IDLE model from ModelBuffer 2406 and calculates the matrix $M(k)$ and the residual $R_{B,M}(k)$, which are both compressed in module CompModRes 2416. The outputs of module CompModRes 2416 and module CompMod 2418, which compresses the current IDLE model are given to the module CombMod 2420, where they are

combined. The combination is the result of a decision process based on the amount of data necessary for the transmission of either the current basis images $I_B(k)$ or both the matrix $M(k)$ and the residual $R_{B,M}(k)$. This decision can also be performed for each factor. The module TraCombMod 2422 transmits this combined model. The module Predict 2408 provides predictions for the images of the SCI. The module 2412 calculates the difference between the original images and these predictions. The resulting residuals are compressed in module CompRes 2414 and transmitted in module TraRes 2424.

The above procedure can be used separately for each attribute independently, e.g. for the vertical motion subspace model, the horizontal one and the subspace model of intensity changes. Alternatively, the procedure described above may be applied jointly on basis images of different attributes.

As described above, the transformation matrix $M(k)$ was computed by projecting the basis images of the current SCI on the basis images of the previous SCI. This is the optimum solution if the sum of squares of the residual is used as the cost function to be minimised. However, it need not be an optimum solution with regard to the task of data compression. Therefore, an alternative cost function for optimisation may be defined, which better represents the relation between the required amount of data and the matrices $M(k)$ and $R_{B,M}(k)$. Such a cost function takes the elements of the transformation matrix $M(k)$ as parameters. An appropriately chosen optimisation method transforms the previous basis images $I_B(k-1)$ according to an initial matrix $M(k)$, and computes residuals $R_{B,M}(k)$ as the difference between the current basis images $I_B(k)$ and the transformed basis images. The cost function evaluates the quality of this step, the optimisation method produces a correction term for $M(k)$ and decides on the further continuation of the algorithm until convergence.

In general, there is motion between the reference images corresponding to consecutive SCIs. This can be exploited for the basis images if the modelling was done in reference position. As an example consider a face says something captured at one image position throughout a set of consecutive images. The same face saying something similar at another image position throughout a subsequent set of consecutive images. The subspace models of motion for the talking head in the two SCIs will have a common or a very similar subspace model of motion. However, the basis images relating to the talking head will be located in different spatial parts of the basis images. In such cases, it may be beneficial to compensate for this difference in position by representing the current basis images $I_B(k)$ as warped versions of the basis images

$I_B(k-1)$. From a decoding perspective the pertaining transformation can be computed before applying the Warp operator

$$I_B(k) = \text{Warp}(M(k) \cdot I_B(k-1), D) + R_{B,M}(k)$$

In this case, the basis images from the previous SCI $I_B(k-1)$ are transformed using a matrix $M(k)$. The result is warped according to the motion field D between the two reference images of the models. To this result, a residual $R_{B,M}(k)$ can be added optionally. For this scenario the matrix $M(k)$ may be determined by

$$M(k) = \text{OppositeWarp}(I_B(k), D) \cdot I_B(k-1)^T \cdot (I_B(k-1) \cdot I_B(k-1)^T)^{-1}.$$

An implementation structure for the functions to be performed inside the module CalcModRes 2410 is depicted in Fig. 24b. The module 2430 receives the preceding IDLE model 2431 and the current IDLE model 2432. The module CalcMotion 2434 calculates the motion field D between the two reference images of the models. This motion field and the current IDLE model are given to the module OppositeWarp 2436, where the basis images $I_B(k)$ of the current IDLE model are warped to the co-ordinate system of the basis images of the preceding IDLE model. The module CalcM 2438 computes the matrix $M(k)$ using the basis images of the preceding IDLE model and the warped basis images of the current IDLE model according to the equation given above. The matrix $M(k)$ is fed to the module MatrixMult 2440 and returned as output 2446. The module MatrixMult 2440 performs the transformation $M(k) \cdot I_B(k-1)$. The result is put to the module Warp 2442 which warps its input to the co-ordinate system of the basis images of the current IDLE model according to the motion field D. The module 2444 calculates the residual $R_{B,M}(k)$ as the difference between the basis images of the current IDLE model and the output of the module Warp 2442.

Similarly, the pertaining transformation can be computed after applying the Warp operator. The further procedure continues along the lines described above as

$$I_B(k) = \tilde{M}(k) \cdot \text{Warp}(I_B(k-1), D) + R_{B,\tilde{M}}(k).$$

For this scenario the matrix $\tilde{M}(k)$ may be determined by

$$\tilde{M}(k) = I_B(k) \cdot \tilde{I}_B(k-1)^T \cdot \left(\tilde{I}_B(k-1) \cdot \tilde{I}_B(k-1)^T\right)^{-1} \quad \text{with} \quad \tilde{I}_B(k-1) = \text{Warp}\left(I_B(k-1), D\right).$$

An implementation structure for the functions to be performed inside the module CalcModRes 2410 is depicted in Fig. 24c. The module 2450 receives the preceding IDLE model 2452 and the current IDLE model 2451. The module CalcMotion 2454 calculates the motion field D between the two reference images of the models. This motion field and the preceding IDLE model are given to the module Warp 2456, where the basis images $I_B(k-1)$ of the preceding IDLE model are warped to the co-ordinate system of the basis images of the current IDLE model. The module Calc$\tilde{M}$ 2458 computes the matrix $\tilde{M}(k)$ using the basis images of the current IDLE model and the warped basis images of the preceding IDLE model according to the equation given above. The matrix $\tilde{M}(k)$ is fed to the module MatrixMult 2460 and returned as output 2465. The module MatrixMult 2460 performs the transformation $\tilde{M}(k) \cdot \text{Warp}(I_B(k-1), D)$. The module 2462 calculates the residual , $R_{B\tilde{M}}(k)$ as the difference between the basis images of the current IDLE model and the output of the module MatrixMult 2460.

A decoder corresponding to the procedure described above has to provide the means to perform the transformation of basis images already resident in memory according to the matrix $M(k)$ or $\tilde{M}(k)$, respectively. Additionally, the decoder must provide the means to perform a warp function on the basis images stored in memory.

The method described for compression according to the present invention in any of its embodiments as described herein can be implemented on a standard computer by a corresponding computer program, as will be readily apparent to an expert. It may also be implemented by a special purpose chip, such as an ASIC, or by a Digital Signal Processor (DSP) performing the method of the present invention. By decoding the encoded data resulting from the compression method of the present invention the sequence of video images which has been encoded may be reconstructed. A corresponding method for decoding may, for example, comprise decompressing, rebuilding motion fields from subspace models, warping images and adding image residuals in order to reconstruct the sequence of video images. The corresponding apparatus may, for example, comprise means to perform the aforementioned steps .

The method may be used to reduce the amount of bits necessary for storing the video sequence on a storage medium, for transmitting it via radio broadcasting, satellite links, optical fibers, or via any communications link, e.g. also through the internet or any computer network. Encoding and/or decoding may be performed either on data written to and/or read from a storage medium, it may also be performed on data transmitted and/or received over any communications link, in this case the decoding may also be performed in real time. The method may also be performed by a device connected to a TV set or a satellite receiver, it may then be performed in

combination with any methods assuring security against unauthorized use such as encryption/decryption methods. The program code which enables a computer to perform the method of the invention may be stored on a floppy disk, in a memory of the computer, or it may be delivered to the computer via any transmission medium like e.g. an internet link.